BEEBLET

$3-00

ISSN 0112-3351

VOLUME 3 NUMBER 2 MARCH 1985

BBC/ACORN

COMPUTER USER GROUP
NZ (inc)

CONTENTS

Editorial 3
Enigma Beeb Solution 4
Speech Utility 6
MENU: Tape indexing program 8
Dicing with Histograms 10
Mail 12
Graphics Window 14
Electron: Plus-3 Interface 15
Find the Actual Colours 16
Software for the Blind 17
Super Dual Beeb 18
Anne's Complaint 19
Bitwise Operator 21
Aventure Development System 27
Student discount 29
Local Arrays 30
Frak - Review 31
Disc Sector Verify 33
Twin Kingdom Valley - Review 34
System Integer Chart 35
Triffid Software Research 36
Fast Fingers Only - Review 37
Cartoon 28
Hall of Fame 39
Local Meetings 40

ADVERTISERS INDEX

Einstein Scientific Ltd 13

****

BEEBLET is edited by Bruce Wills who may sometimes be tracked down on 861-662 (h) or 720-030 (wk).

DISCLAIMER

The views of the reviewers, the Editor and other contributors are not necessarily shared by the BBC/ACORN COMPUTER USER GROUP OF NZ INC.

EDITORIAL

Please, could members include their MEMBERSHIP NUMBERS on all correspondence.

****

Advertising revenue made a tidy contribution to the magazine's budget last year. Over the last few months there has been little interest from the business community. Why? : "...and we only seem to sell one or two copies of each program per centre..." SOFTWARE PIRACY is thus hurting all of us indirectly. It has the effect of forcing up unit costs and prices throughout the industry. Sooner or later someone has to pay for this, and if software companies are to stay in business, that someone will be the customer - you and me. Please, next time you are tempted to copy a commercial program, THINK about the collective impact of this practice.

Next month I will get a commercial viewpoint: does anyone know of someone in the industry keen to write a guest editorial (..Barsons...?)?

****

Most of you will by now be aware of the recent difficulties encountered by Acorn and the subsequent aquision of a substantial part of its shareholding by Olivetti. The corporate structure of Acorn has been revised and Doug Pauling (Barson's Managing Director) has expressed confidence in the new organisation's continuing support and development of the BBC (and presumably the Electron which is rapidly developing into a very attractive system). The BBC has also renewed its contract with Acorn to produce a 'BBC' computer.

****

There has been a disproportionate contribution to the magazine by a few members - Anne has opened up with her big guns inside. Please, don't be upset, if you are one of the many people who have sent in material - especially if a reply or other action is caught up in my current backlog. You can expect your tapes, discs etc back in the near future as I move this. However, unless I get a lot of material fairly soon, the April edition could be a bit of a sad joke - suppose I could fill it up with AGM minutes etc. I always welcome letters which indicate what people do, or do not want, especially if they are specific. Remember however (Gavin & Anne apart) that if the magazine does not contain things which interest you, it could be because YOU haven't supplied them. Or twisted someone else's arm...

ENIGMA BEEB SOLUTION

G.C.F.

I make no claims that this is the fastest solution, but it does not rely on any cunning deductions to simplify the problem. Only two very obvious deductions are made:

a) that zero is not a divisor,
b) that AA is the same as 11*A

The program does not even assume that the solution is unique; it actually prints only one solution along with the time in seconds it took (under 6 minutes)

Although many compromises have been made for speed, the method used should not be hard to follow. The program constructs every number which has 6 different digits, and tests them against the divisors one by one.

CODING NOTES:

These all apply to the monstrosity of line 370.

a) Using complex IF statements in Beeb Basic is usually (?!) a sure road to madness, and best avoided.
b) Always leave a space between MOD and any variable beginning with "E" - MODE% is a syntax error!
c) Readers may be unfamiliar with the fact that "logical expressions" can have "arithmetic values" eg
PRINT(6=6) gives -1,
PRINT(6=5) gives 0,
PRINT(6=6)+(5=5) gives -2

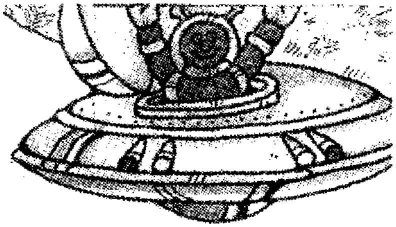
FINAL COMMENT:

The listing here was printed immediately after the program had run successfully. Do not type in any spaces between thr line number and the first statement on each line (U.G. p.290)

* * *

The 'Elite' game appears to have made a major hit judging by the number of Hall of Fame entries I received over the last couple of months. Joanne C. (Elite status) reports a score of 1,100,300 credits on the cassette version and 120,000 credits and two tasks performed on the disc version. Joanne wants to know whether anyone has more than two tasks done.

```
  10 REM  ENIGMA
  20 TIME=0
  30 DIM R$(6)
  40 DIM Q$(6)
  50 R$(0) = "987654321"
  60 Q$(0)=""
  70 FOR R% = 1 TO 9
  80   E% = FNa(1,R%)
  90   FOR S% = 1 TO 8
 100     N% = FNa(2,S%)
 110     FOR T% = 1 TO 7
 120       I% = FNa(3,T%)
 130       FOR U% = 1 TO 6
 140         G% = FNa(4,U%)
 150         FOR V% = 1 TO 5
 160           M% = FNa(5,V%)
 170           FOR W% = 1 TO 4
 180             A% = FNa(6,W%)
 190             Q% = VAL(Q$(6))
 200             IF Q%MOD11=0 THEN PROCb
 210             NEXT
 220           NEXT
 230         NEXT
 240       NEXT
 250     NEXT
 260   NEXT
 270 END
 280
 290 DEF FNa(X%,Y%)
 300 T$ = R$(X%-1)
 310 D$ = MID$(T$,Y%,1)
 320 Q$(X%) = Q$(X%-1) + D$
 330 R$(X%) = LEFT$(T$,Y%-1) + MID$(T$,Y%+1)
 340 =VAL(D$)
 350
 360 DEF PROCb
 370 IF Q%MOD E%=0 IF Q%MODN%=0 IF Q%MODI%=0 IF Q%MODG%=0 IF Q%MODM%=0 IF Q%MOD
A%=0 IF (Q%MODVAL(LEFT$(R$(6),1))=0)+(Q%MODVAL(MID$(R$(6),2,1))=0)+(Q%MODVAL(RIG
HT$(R$(6),1))=0) = -2 THEN PRINT Q%,TIME/100
 380 ENDPROC
```



SPEECH UTILITY

David Baker

Application:

The Acorn Speech Synthesizer PROM is normally activated by the SOUND-1 statement, to vocalise words or word_parts stored in its memory, when a program is running.

However due to my poor sight (I am a registered blind person with 3% sight), I had a need to check information input from the keyboard instead of the usual visual monitor check. I am fortunate that my wife will normally help by 'calling' the program as I enter it, and check the VDU, but on those occasions when I am doing a solo, I tend to have many errors to de-bug later, just due to typing mistakes!

I wish to acknowledge the assistance of Frank Hazlewood (a fellow club-member), who developed and wrote the following program, so that I am able to use the computer with greater ease.

In fact the program has proved far more successful than originally envisaged, as I have discovered several different applications for the program, other than the one originally designed for. Tests show that the speech facility should be switched off (f1 key) before RUNning the program below, or it will be corrupted and be processed to the screen in "slow motion" lumps of data. However this "slow motion" effect can be put to good use if you want the computer to read back when you request LIST; this saves me peering at the screen for long periods.

WHAT IT DOES:

The program intercepts a character which is about to be printed on the screen, diverts it to the speech processor, then forwards it to the screen for display.
The red user-definable keys: f0 turns the facility on, and f1 turns it off.

HOW TO SET IT UP

a) Type in the program in the normal manner.
b) SAVE before proceeding.
c) RUN <RETURN> The program will be assembled on screen.
d) enter: PRINT ~TOP <RETURN>
The computer will respond with a hex number. (call it xxxx for now)
e) enter PAGE=&xxxx+1
this step prevents its being overwritten by a following program or other entries.

(Ed.. A possible alternative - add an extra line at the very end

of the program which reads:
PAGE=TOP+1   untested, so over to your experiment.)

f) Press f1 key to initiate speech production. Entries will now begin from the new value of PAGE.

```
   10 REM....."KEY-SYNTH"..........
   20 REM..for Acorn Speech PROM...
   30 REM.....By F.Hazlewood.......
   40 REM.....July 1984............
   50 REM.....(for D.Baker)........
   60  *KEY 0 "?&020E=(Q% MOD 256):?&020F=(Q% DIV 256) M"
   70  *KEY 1 "?&020E=&A4:?&020F=&E0 M"
   80 DIM Q% 100
   90 DIM H% 8
  100  FOR Z%=0 TO 3 STEP 3
  110  P%=Q%
  120 [OPT Z%
  130 STA H%+2
  140 PHA
  150 TXA
  160 PHA
  170 TYA
  180 PHA
  190 LDA #&FF
  200 STA H%
  210 STA H%+1
  220 LDA #&00
  230 .LOOP STA H%+3,X
  240 INX
  250 CPX #&05
  260 BNE LOOP
  270 LDA #&07
  280 LDX #(H% MOD 256)
  290 LDY #(H% DIV 256)
  300 JSR &FFF1
  310 CLC
  320 PLA
  330 TAY
  340 PLA
  350 TAX
  360 PLA
  370 JMP &E0A4
  380 ]NEXT Z%
  390 ON ERROR NEW
  400 NEW
```

MENU: a program for indexing tape

David Baker

Although I am new to computing, counting in months, by now I have started to accumulate a number of different programs of various types, which I have either copied from magazines or received by other means (eg the club's introductory tape).

Early on I decided that I should make "back-up" copies of all programs for safety purposes. I put all these onto long cassettes (eg C60 or C90), but as these tapes can hold a good number of programs, I needed an easy index to what was written. I was not happy with the idea of putting a piece of paper with written details into the cassette lid, as it may be lost or become tatty etc with age. To *CAT a tape would still not tell me tape counter positions.

I therefore decided that if disc users could have their DIRECTORIES (and equivalents), so could tape users. The MENU program here is recorded on to the beginning of the tape, and the user begins the subsequent programs well into the tape to allow for possible expansion of the MENU program as entries are made on it.

With the index right there on the tape it can't get lost, and it's very convenient to LOAD and RUN for ready reference, and you can't lose it (unless you lose the tape!).

You will need to enter the program name and tape counter position on the appropriate line of the program for each one. I have left room for a third column of information to be added - eg Game, Utility etc, or program block lengths - whatever you might think useful. Each time you add more programs on the end of the tape, you will need to reload MENU and add these items, then SAVE this new version. This is why you must leave a decent gap between MENU and the first program on the tape, because each time you change MENU it will get bigger.

```
 10 REM......"MENU" for backup tape indexing..
 20 REM......by D. Baker, 1984..............."
 30
 40 MODE6
 50 PRINT"MENU"
 60 PRINT"................................."
 70 PRINT"Side One-Page One"
 80 PRINT"................................."
 90 PRINT"   Program:    Counter:"
100 PRINTTAB(0,6)"1"
110 PRINT"2"
```

```
120 PRINT"3"
130 PRINT"4"
140 PRINT"5"
150 PRINT"6"
160 PRINT"7"
170 PRINT"8"
180 PRINT"9"
190 PRINT"10"
200 PRINT"11"
210 PRINT"12"
220 PRINT"13"
230 PRINT"14"
240 PRINT"......................................"
250 PRINT"    Press (SPACE) for next page."
260 PRINT"    Press (ESCAPE) to load next program."
270 PRINT"......................................"
280 REPEAT:UNTIL GET=32
290 CLS
300 PRINT"MENU"
310 PRINT"......................................"
320 PRINT"Side One-Page Two"
330 PRINT"......................................"
340 PRINT"    Program:    Counter:"
350 PRINTTAB(0,6)"15"
360 PRINT"16"
370 PRINT"17"
380 PRINT"18"
390 PRINT"19"
400 PRINT"20"
410 PRINT"21"
420 PRINT"22"
430 PRINT"23"
440 PRINT"24"
450 PRINT"25"
460 PRINT"26"
470 PRINT"27"
480 PRINT"28"
490 PRINT"......................................"
500 PRINT"    Press (ESCAPE) to load program."
510 PRINT"    END OF SIDE ONE MENU."
520 PRINT"......................................"
530 REM....add more 'pages' if necessary!
```

DICING WITH HISTOGRAMS

G.C.F.

INTRODUCTION:

Ever wanted to throw a couple of 3-sided dice? Sorry, that's impossible! But the computer does not "know" that, and will be quite happy to accept any dice with at least 2 sides, and will cope with more than 100 sides.

Teachers may like their students to predict the shape of histograms for various combinations of numbers of dice and numbers of sides. Budding mathematicians might like to derive the expression (found in line 190) for the maximum number of dice allowed if the maximum number of possible results (size%) is not to be exceeded, given the number of sides (S%) per dice - eg 2 dice each with 6 sides will produce 11 results, from 2 to 12.

Much to my disgust, getting this program correct took much more trial and error than I would like to admit! It is now as robust as I can make it, without going to extremes (to disable ESCape you could add *FX229,1 I suppose).

PROGRAM COMMENTS

People with either tvs or Model As might like to change mode% to 4. The range of values the program can usefully display depends on the width of a pixel (x-pixel%), which in turn depends on MODE (mode%). The program will ask for the numbers of dice and sides to be within specified ranges. It also insists that the dice are to be "thrown" at least once, but cannot determine in advance the maximum permissable number of throws, which depends on 'chance' (!) - however it will abandon the histogram if it gets too big. The histogram is automatically scaled to fit the entire width of the screen, so that the minimum and maximum possible values will be at extreme left and right respectively.

```
10 REM  DICEHST
20 REM  Simulates the throwing of various numbers of different types
30 REM  of dice and draws a histogram showing how often each total occurs.
40 REM  N.B.  It allows 3 sided dice to be used !!!!
```

*FX LIBRARY

John Baker sent in the following *FX calls not documented in the U.G.

*FX210,x (where x is 3 or more) : turns sound off

*FX210,0 : turns sound on.

```
 50 mode% = 0
 60 MODE mode%
 70 x_pixel% = 2^(1 + mode%MOD3) :REM valid for modes 0, 1, 2, 4 and 5
 80 size% = 1280/x_pixel%
 90 DIM tally%(size% - 1)
100 REPEAT
110   REPEAT
120     PRINT "Number of sides [2...";size%;"] ";
130     INPUT S%
140     ok% = TRUE
150     IF S% < 2 THEN PRINT"Must have at least 2 sides": ok% = FALSE
160     IF S% > size% THEN PRINT"Too many sides!": ok% = FALSE
170     UNTIL ok%
180   REPEAT
190     PRINT "Number of dice [1...";(size% - 1)DIV(S% - 1);"] ";
200     INPUT N%
210     ok% = TRUE
220     IF N% < 1 THEN PRINT"Must have at least one dice.": ok% = FALSE
230     R% = (S% - 1)*N% + 1
240     IF R% > size% THEN PRINT"# dice too big for the # sides": ok% = FALSE
250     UNTIL ok%
260   REPEAT
270     INPUT"Number of throws "M%
280     UNTIL M% > 0
290   CLS
300   PRINT ;M%;" throws of ";N%;" dice each with ";S%;" sides."
310   scale = 1280/R%
320   FOR I% = 0 TO (R% - 1)
330     tally%(I%) = 0
340     NEXT I%
350   throw% = 1
360   REPEAT
370     C% = 0
380     FOR D% = 1 TO N%
390       C% = C% + RND(S%)
400       NEXT D%
410     bar% = C% - N%
420     T% = tally%(bar%) + 1
430     tally%(bar%) = T%
440     MOVE scale*bar%, 4*T%
450     DRAW scale*(bar% + 1) - x_pixel%, 4*T%
460     throw% = throw% + 1
470     UNTIL throw%>M% OR T%>247
480   IF T%>247 THEN PRINT"(HISTOGRAM ABANDONED"'" AFTER ";throw%;" THROWS)"
490   UNTIL FALSE
500 END
```

MAIL

Another letter has arrived from the intrepid John B. of Waiheke Island..
From John B to Neil B. - Greetings. Re: "to much chat not enough listings". Point taken, enough is enough. However, take heed:-

These pages were made for listings
And thats just what they will give
But beware, one of these days all this listing
May take more than it can give

My apology to Nancy Sinatra.

Can one of you bright boys (or ladies - Ed.) give this old codger a little programming help. My problem is as follows. One line in the listing I am typing calls for the character [(assembly language). The. Electron doesn't have a key for [or]. The Electron guide book gives VDU 91 for the character [and VDU 93 for the character]

The listing I am referring to includes:-
Line 40 [OPT T
If I type in VDU91 [OPT T I get "NO such variable at line 40"

I have solved this problem in a rather messy way. I type VDU 91 and get on screen [>. I then get the edit cursor under [(press copy) type (line no.) [OPT T press return. And the program runs.

There must be an easier way out. I ask myself why I kept the Electron for myself. My answer is that it is a challenge to me. It prevents atrophy of my brain cells!

****

David Baker has also made some interesting comments in the letter accompanying his contributions. Picking up Anne's point about straight-forward programs more useful to the average member, David says "To encourage other 'greenies like myself who may be overawed, confused or disheartened by the high tech jargon floating around in the magazine, I have submitted my first attempt at simple programming in the form of the "MENU" listing."

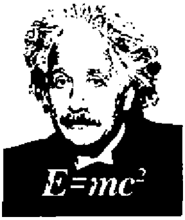
I am sure that many other U.G. members will have written simple but useful, interesting and/or entertaining programs. This is the place to get them published!

EINSTEIN SCIENTIFIC LIMITED
177 Willis Street
WELLINGTON.

CHECK EINSTEIN'S EXTENSIVE RANGE OF PRINTERS AND SOFTWARE FOR THE BBC & ELECTRON MICROS.

WELLINGTON
177 Willis Street. Phone Raju, Michael or Malcolm 851055 or 844353.
PALMERSTON NORTH
154 Broadway Avenue. Phone Richard 64108.
NEW PLYMOUTH
Corner King & Egmont Streets. Phone Bill or Sally 85528.
CHRISTCHURCH
Shop 41, 1st floor, Cashfields Mall Phone Mike 66442.
HAMILTON
inside DIC, Garden Place. Phone Maxine 81969.

Einstein
SCIENTIFIC LTD



The computer world made simple.

GRAPHICS WINDOWS

Did you know that the fastest way to colour in a rectangle is by defining it as a graphics window? It's faster than drawing two triangles, and doesn't have the diagonal line through it that often occurs with the PLOT85 method. It's also faster than filling using PLOT77.

A graphics window is also useful when things are happening which you don't want to interfere with some text or other picture on the screen - eg prompts in your drawing package program. Electron Owners could look again at the Sketch program which is on their Welcome Tape. Another use is for cheating - you have PROCdraw_shape that you don't want to change (someone else wrote it; you can use but not understand it), but you want only part of the shape it draws. You can define a window, and draw the shape with co-ordinates that ensure part of it is of out of the boundaries of the window, then reset the window to a more useful size.

Perhaps you are in an underground cave system or a maze. The program may well draw the entire map each time, but what you can see is limited by the light of your lamp. A graphics window can be set up so that although the whole map may be drawn each move, only the area you can see will actually appear on the screen.

An example program:

```
 10 MODE2
 20 PROCdraw_grid
 30 WAIT=INKEY(100)
 40 CLS
 50 VDU24,700;100;1100;700;
 60 CLG
 70 PROCdraw_grid
 80 VDU26
 90 END
100
110 DEF PROCdraw_grid
120 GCOL0,3
130 FOR I%=1 TO 1000 STEP 120
140   MOVE I%,0:DRAW I%,1023
150   NEXT I%
160 FOR I%=1 TO 1280 STEP 120
170   MOVE I%,0:DRAW 1279,I%
180   NEXT I%
190 ENDPROC
```

THE ELECTRON PLUS 3 DISC DRIVE

H. Wiggins

It is always nice to get something new for your computer and here is a real honey. The plus 3 is an L shaped package in matching trim which bolts firmly on the back of the Electron. Together with the Plus 1, which again bolts to the rear, you have a package which contains a disc drive, a printer interface and an A to D converter. It has nice transportability without too many cables. The discs are 3.5 inch minifloppies, double density, single sided, 320K. It is said that these were chosen because of their compactness, improved technical features, and general industry acceptance. Some support to this view is given by the fact that spare discs were labelled "Hewlett Packard". The demonstration disc was full of goodies. First the sample program loaded instantly and displayed instantly on the screen, compared with the slow building up of the picture in the tape version. The Advanced Disc Filing System (ADFS) does not bely its name and Acorn have taken a lot of trouble over this one. It is not compatible with the BBC DFS, but contains some 34 commands.

Then there is the Utilities menu. Some of these are:

*EFORM	Formats a disc so that it may be used with the ADFS
*VERIFY	Checks that a disc's contents are not corrupt
*BACKUP	Copies the entire contents of one disc to another
*LIST	Displays a text file with line numbers
*TYPE	Displays a text file without line numbers
*BUILD	Creates a text file from the keyboard
*DUMP	Displays a file in hex and ASTI-15
SETPARAMS	Sets up various ADFS options
UTILS	Produces the menu of utility options.

A HELP command is available to explain these ultilities.

Familiar commands in BASIC such as SAVE, LOAD and CHAIN are not lost. The ADFS can be disabled with NOADFS so that normal facilities are there. LOading from and to tape to ADFS is simple. However, there are space problems since PAGE is normally at 1D00 and some tapes give DIM or MODE errors. There are ways of reclaiming space and a local agent is compiling a list. HIMEM - LOMEM gives 20990 with NODFS and 17150 with (MODE 6). In MODES 0, 1 and 2 the figure becomes 4862 so this explains the problem. The BBC has a similar problem.

I have still not found out all the things I have bought.. an outlet is provided for a second disc drive 5.25 or 3.5 inch.. The manual requires patience, however I expect the usual good amount of supporting literature to become available. I have had my Electron some 9 months and have found it a stimulating companion. With this new facility it is a new and powerful ball game. The Acorn has grown.

FIND THE ACTUAL COLOUR

GCF

If you are writing a complex and colourful program (game?) which keeps redefining what the logical colours mean by using VDU19, and you keep calling a PROC which simply must know what the current actual colour is, then this program could be of use. The function below could be used if your PROC has to temporarily redefine a logical colour for its own purposes but must also restore it to its original assignment when it is finished.

N.B. This program uses a "spare" byte in Page Zero to avoid having to set up a PARAMETER BLOCK by a DIM statement. This means that the Function can be copied into any other program without having to worry about setting up anything special in advance. If you are already using this location, then you know enough to find a spare byte somewhere else!

```
 10 REM 'DEMO' of reading colour palette.
 20 REM (OSWORD with A=&0B)
 30 REM See USER GUIDE pages 458, 462 & 463
 40 FOR mode%=0 TO 2
 50   MODE mode%
 60   FOR A%=0 TO 15
 70     FOR L%=0 TO 2^(2*mode%)-1
 80       VDU19,L%,A%;0;
 90       PRINT L%,FN_actual_colour(L%)
100       dummy%=INKEY(200)
110       NEXT L%
120     NEXT A%
130   NEXT mode%
140 END
150
160 DEF FN_actual_colour(logical_colour%)
170 LOCAL A%,M%,X%,Y%
180 M%=&70
190 A%=&0B
200 X%=M%MOD256
210 Y%=M%DIV256
220 ?M%=logical_colour%
230 CALL &FFF1
240 =?(M%+1)
```

SOFTWARE FOR THE BLIND

With the advent of the speech system fitted internally to the BBC, St Dunstan's, an organisation in England for men and women blinded in the war, was prompted to sponsor a research project to ascertain whether a totally blind person could use a computer to enhance their quality of life.

As an aid to living, leisure and communication, "the talking BBC micro replaces braille for the storage and quick retrieval of data" says St Dunstan's. Although the vocabulary is limited to some 160 words, and the speech output is in serial form, its usefulness has been demonstrated by the following programs which were specially written for the project. A VDU is unnecessary, but reference is made to "screen" to simplify the description of the various programs below:

PRINTREAD: characters are spoken as they are printed to screen either from keyboard input or generated by the computer.
SCREENREAD: lines of text previously printed on to the screen can be selected and 'read'. This program also incorporates the "Printread" program.
VOCAB: this is a data retrieval program enabling the user to find a particular whole "spoken" word or word number, and insert it into a program immediately.
DATAFILE: an information storage and retrieval program for the user to establish an address book, telephone directory, index of various items (eg record collection), record of personal confidential information etc. Used with the Printread program, which is loaded first. This 'paperless' system is suited to the needs of those who cannot get sighted help.
TEXT10: A program for the 'Acorn' speech synthesiser; a simple text-editing program enabling the user to compose text on the computer and screen, and amend before getting a printout.
MVTEXT: the 'Microvox' speech synthesiser version of TEXT10
CW: a morse code program, with several options. The user has the choice of selecting the random character send mode, text send mode, typing from the keyboard, or recording tapes into memory. The user can also define the send speed and delay period between characters.
CLOCK: will tell the time when a key is pressed.
Two further programs for Dominoes and Cribbage are in various stages of completion.

If any User Group member should know of anyone (particularly a handicapped person) who would benefit from obtaining copies of these programs, please contact me direct, sending a blank tape and stamp (plus a longer tape if requiring the TEXT programs for spoken instructions and demonstration):

David Baker, 67 Bayview Rd, Glenfield, Auckland 10 or phone me (pvt) 444-2122

SUPER DUAL BEEB

GCF

DEFINITION:
A Beeb with an active 6502 co-processor and dual disc drive system (2 drives and 2 processors!)

CHARACTERISTICS:
People suddenly start referring to the box formerly called "Beeb" as the "I/O processor"!!

SPEED

Although the co-processor manual claims "up to 50% faster processing", some graphics programs will run at more than twice their normal speed - while the I/O processor is drawing a line or filling a triangle etc, the Basic program continues in the Language processor (2nd 6502 box?) until it has to wait for the I/O processor to finish its last request.
This means that with the right mix of calculations and graphics, you can get close to the performance of a single 6502 running at 5MHz! (2+3 MHz). Space Invaders anyone? An actual, highly tuned benchmark program gave close to this performance with a nominal equivalent of 4.9MHz. In practice, the speeding up of programs will be uneven - depending on the balance of work done by each processor, which will vary in different parts of a program.

Disc I/O

Super dual Beebs have the 1.2 DFS a part of the DFNS. It is claimed that this results in 20% faster disc accesses. Probably more significant is that *COPY and *BACKUP do NOT corrupt a Basic program still in the language processor! This means that if you get "disc full", it is possible to *COMPACT, THEN save the program! This was not possible if the program was running in the Beeb without active co-processor - you had to scrabble among the heaps of discs under the telly for one with some space on it.

Memory

Naturally "everyone" knows that the co-processor allows 44K for a Basic program irrespective of Mode. But how many know that there is at least 4.5 K in the I/O processor that could be used by an enterprising assembler programmer? The area is between &1F00 and &3000 on a disc system in a 20K graphics mode with space set aside for redefining all the printable characters.

ANNE'S COMPLAINT:

WHAT IS A USER GROUP?

1) A group of people having something in common - eg computers of a specific type - who want to share experiences (as in the Constitution)

or

2) A group of people who mainly sit back and sponge off the attempts of a few to share what they have discovered, or help where they can.

'Beeblet' March'84 to Feb '85 pages were made up in this way:

11.5% written by Anne
8.5% written by Gavin

That's 20% of the mag written by two people! (as we're married to each other, you could argue that in fact it's only from 1 person!)

10% of all pages was taken up by covers, editorial and Contents
5.8% written by Anton
5.2% by Neville T
4.4% by Warren W
3.9% by Tony K
3.5% by Robert N

That's over 50% of the magazine from 8 people including the Editor who puts it all together.

47.2% was supplied by less than a further 80 people (excluding the Hall of Fame)

It would appear that over 80% of the User Group have made no contribution to the magazine, particularly to the detriment of those who do not live near places holding local meetings. Five of the above are or have been members of the National Committee; only one comes from Auckland, and one from Christchurch. Surely those who have access to meetings should make particular effort for those who don't, to help them too in the only way possible - through the magazine.

WHAT about all those UNANSWERED calls for HELP in the magazine? To assume "someone else" will supply the answer is demonstrably and frustratingly wrong. Very few such cries have been attended to.

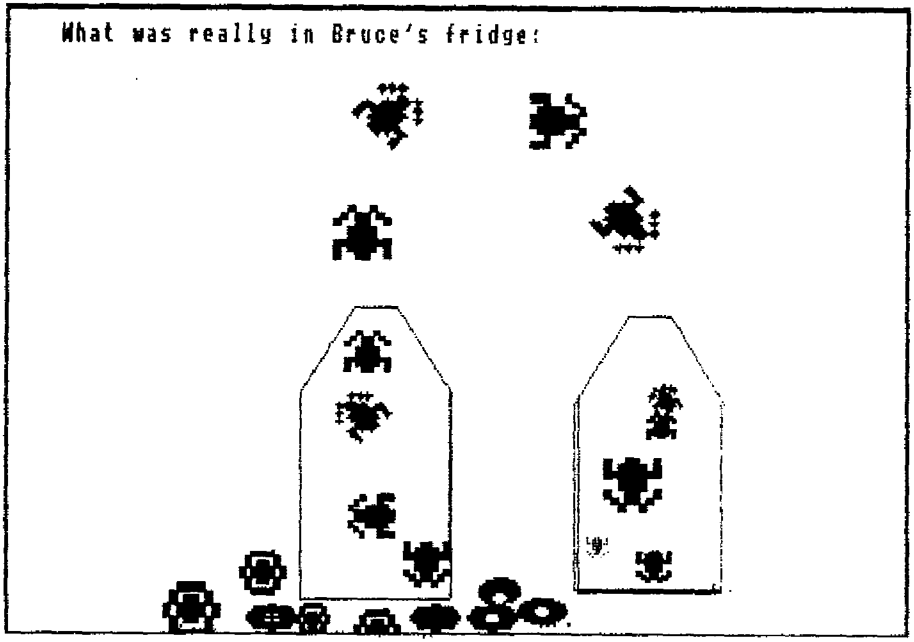
IF you want a chance to get some free goodies, ENTER the competitions! It seems that if you enter you are guaranteed to be a winner!

The only way to be sure you won't read an article you don't want is by supplying the Editor with YOUR alternative for those pages.

HELP WANTED

1) Give ME some incentive to open the pages of Beeblet in the coming year - 'Beeblet' is boring - I've read it all before. So has my husband. Thoroughly.

2) Read through previous Beeblets. There are cries for help still crying, including some of mine.



What was really in Bruce's fridge:

BITWISE OPERATORS

Anton E.

BASIC KEYWORDS: AND, OR & EOR

The BBC User Guide is explicit in its description of these operators, but does not deal with them as a whole, nor does it give their uses. This article will describe how they work, and what they can be used for.

The term "bitwise" comes from their mode of action. If we take a number, say A%, comprising 32 bits, and operate with a second number, say B%, then each bit of A% interacts with the corresponding bit of B%, and has no effect on adjacent bits. (The usual arithmetic operators (+-/*) involve carry over from bit to bit.) This is the entire basis of their value. They enable very selective changes in bit settings to be made, without complications between adjacent bits.

We will only discuss operations between integers here, as "reals" act similarly, but in a confusing way, with no advantage. You should also realise that in actual fact, the "integer" (eg. A%) is a collection of yes/no answers that can mean anything you want, and in many cases, mean nothing at all since some "answers" may have no question. One important operation I will explain is the masking off of unwanted bits, by setting them to zero. Bits can't be removed, but they can often be ignored if they have a zero state.

The three operations wanted are a way of setting bits (to 1), clearing bits (to 0) and toggling bits. (1 to 0 and 0 to 1)
AND, OR & AND can be used to do this, but not in a straightforward manner.
Here is a list of effects:

For any matching pair of bits:

(0 AND 0) = 0   (0 OR 0) = 0   (0 EOR 0) = 0
(0 AND 1) = 0   (0 OR 1) = 1   (0 EOR 1) = 1
(1 AND 0) = 0   (1 OR 0) = 1   (1 EOR 0) = 1
(1 AND 1) = 1   (1 OR 1) = 1   (1 EOR 1) = 0

AND....Only if both are 1, is the result 1
OR.....If either or both are 1, the result is 1
EOR....If the bits are different, the result is 1

NOTE: order has no effect. A% operating with B% is the same as B% operating with A%. Often in the examples that follow, I'll use the order "data followed by mask" but that is my own convention, so I know which number is which. The computer wont care.

To clear a bit, AND it with 0. To set a bit, OR it with 1. To toggle a bit, EOR it with 1. I'll now discuss that in more detail.

Let us now regard A% as the set of 32 bits recording data, and call it data%. We want to copy certain bits of interest into another integer, called somebits%. This is done using the operator AND with an integer, mask%. Any bit of mask% that is 0 will make the same bit in somebits% also 0. But where mask% has a 1, the corresponding bit in somebits% is a copy of the data% bit. Overall, then, somebits% is a copy of data%, but with 0 wherever mask% had a 0.

Use of AND: to clear bits.
The ASCII numbers are carefully designed. All letters of the alphabet have bit 6 set, and the lower case also has bit 5 set. (In numbers, all letters are valued 64 or more, and lower case 96 or more.) To go from upper to lower case, clear bit 5. So, if a program had a GET or a INKEY input line, and you wanted to test for a "Y" of upper or lower case, one way is:

```
1000 reply% = GET
1010 IF (reply% AND 223) = 89 THEN PROCyes
```

The mask (223) has bits 0 to 4, and 6 and 7 set, so any input has those bits retained, but bit 5 is cleared. As bit 7 is not set for any ACSII value, that bit too could be clear, giving a mask of 95 as a valid alternative to 223.

Use of OR: To set bits.
From the operator effects list, you can see that OR tends to set bits. So to copy data% into result%, except for some bits which are to be set, use:

```
    result% = (data% OR mask%)
```

Any bit of mask% that was set will now also be set in result%. If mask% is 32 then bit 5 is set, and if mask% is 1040 then bits 4 and 10 are set. As an example of setting and clearing, the following program goes through the alphabet, printing upper case, setting bit 5 to give lower case, and after printing that, clearing bit 5 and reprinting upper case.

```
10 REM Upper to lower to upper
20 FOR letter% = 1 TO 26
30   ascii% = 64 + letter%
40   PRINT CHR$(ascii%);
```

```
50  lowcase% = (ascii% OR 32)
60  PRINT CHR$(lowcase%);
70  upcase% = (lowcase% AND 223)
80  PRINT CHR$(upcase%); "  ";
90  NEXT letter%
```

In summary, AND clears bits, and OR sets bits. As both operators leave bits in a known state, they can be said to create information, since the result is meaningful. On the other hand, the previous state of the bits has been destroyed, so the operators delete data. It rather depends on whether the result or the original contains more meaning. By saying above "copy data selectively into result", both the original and the result are available. This is a good general rule: If you can afford the space, copy data for program use, and preserve the original. Naturally, if the original state has no meaning, as when setting data bits for the first time, then there is nothing worth preserving. In such cases, one uses statements of the sort:
data% = (data% operator mask%)
But it does pay to preserve, even when you dont think the original will be useful.

The third operator, EOR, called exclusive-or, is different. It toggles bits, and neither creates nor destroys, since two toggles leaves a bit in its untoggled state. If again we use data% as our original, mask% as the bit selection mask, and put the result in result%, then the following rules are useful.
If mask% is zero, (all bits = 0), then result% is an exact copy of data%.
If mask% is -1, (all bits set), then the ones complement number is obtained. That is, all bits of data% are in their opposite state in result%. And if mask% is a mixture of 1's and 0's, some bits are toggled and others are not.

EOR can be used either as a toggle or as a matching method.

The use as a toggle stems from the ease with which (result% = 0) can be tested. Suppose we want to test if bits 4, 7 and 10 in data% are all clear. First we need a mask with bits 4, 7 and 10 set. That's 1168 (16 + 128 + 1024). Use this mask to clear all the other bits, and preserve only the three wanted bits of the original:

```
    result% = (data% AND 1168)
```

Then if result% is zero, the three test bits must all be zero. But now suppose we want to test if bits 4, 7 and 10 were all set. The answer is to toggle the test bits, then test for zero.

```
    result% = ((data% EOR &FFFF) AND 1168)
```

The toggle mask used can be any number that has bits 4, 7 and 10 set. The one I used has all bits from 0 to 15 set. This use of &FFFF

(decimal 65535) has two advantages. Firstly, its easy to remember: hex F means 4 set bits. Secondly, the same mask will work for all cases where up to 16 bits are in use. So if you later change the clearing mask (1168), the toggle mask can stay unchanged.

If you wish to work easily with these masks, the using hex values makes life easier. Hex requires you to have a table of patterns, but almost all arithmetic is eliminated!

0	0000	4	0100	8	1000	C	1100
1	0001	5	0101	9	1001	D	1101
2	0010	6	0110	A	1010	E	1110
3	0011	7	0111	B	1011	F	1111

The 4, 7 and 10 bit pattern is:

```
bit 15.............to 0
    0000 0100 1001 0000
 hex  0    4    9    0
```

Replace 1168 with &490. Note: no adding up needed!
The same hex approach is equally useful in graphics, when defining characters.

A rather unusual use of the toggle is for encoding. Say the Wanganui computer was transmitting information to police terminal no. 4321. A simple fast code is for the main computer and the terminal to share a 256 byte random code number. Each set of 256 bytes sent is first EORed with the code number. The receiver repeats the EOR operation and so decodes the message. Each terminal can have its own code, making interception of transmissions or logging on under an assumed terminal no. quite useless.

EOR is also used for matching bit patterns. If both data% and mask% have the same bit pattern, the result is zero. Not much different from saying "IF data% = mask% THEN ....". Except if data% and mask% have nearly the same pattern, the result has very few bits set and these can be counted. Useful when you have to settle for close matches.

Let us assume we are matching up two bit sets, tuatara% and lizard%. The bits in each set have been set or cleared, in response to questions such as "Is it cold blooded?" (Set bit 5 if yes) Now we want to see how many bits match. (I'll cheat and replace the 32 questions with RND.)

```
10 REM Exclusive-or Matching
20 tuatara% = RND: REM Fake data
30 lizard% = RND: REM Fake data
40 match% = (tuatara% EOR lizard%)
```

```
 50 PRINT ~tuatara%;" EOR ";~lizard%;" = ";~match%
 60 REM First check bit 32, pos/negative
 70 IF match%<0 THEN count% = 1 ELSE count% = 0
 80 REM Then check the other bits
 90 mask% = &40000000: REM bit 31
100 FOR sweep% = 1 TO 31
110   IF (match% AND mask%) THEN count% = count% + 1
120   mask% = mask% DIV 2
130   NEXT
140 PRINT count%; " bits do not match"
```

count% is the number of differences found. The lower the count, the closer match. It would probably be a matter of experience to judge how low a count was significant. Naturally, the computer would be stepping through checking hundreds of entries. lizard% would in an array as, say, beast%(432). This article assumes you have or soon will be able to manipulate arrays. They are more important to understand than bitwise operators!

Well, that's the essence of it. There are a couple of minor points that I should mention. Firstly, operator precedence is crucial. Look up the chapter in the User Guide on it. The important point is that "=" is evaluated before "AND" and that before "OR" and "EOR". Take the example:

```
 IF result% = data% OR mask%
```

The computer evaluates this as:

Create a temporary integer in memory.
Put the answer to result% = data%? there. (Zero if false)
OR the answer with mask%
If overall result is zero goto next line.

Unless mask% is zero, it will set a bit in the answer and result in a TRUE execution of the line. Whereas if brackets are placed round (data% OR mask%), then the first step is to place the result of the OR operation in the temporary location. Then it sees if result% is equal to it. Now the answer will often be FALSE. As you want a high level language to reflect intuitive English usage, PUT BRACKETS IN WHEN USING BITWISE OPERATORS.

The second point is really only a hint. Stick to yes/no questions as far as possible when setting up the data bits. Otherwise ambiguous answers will confound the data search. So "NZ citizen?" is a good question, as either you are or you aren't. But "Is she tall?" doesn't have a precise answer. One way out is to use two bits for each question, giving an either, yes, no and neither option set. So to encode height, (binary rep.), tall = 10, short = 01, middling = 11 and

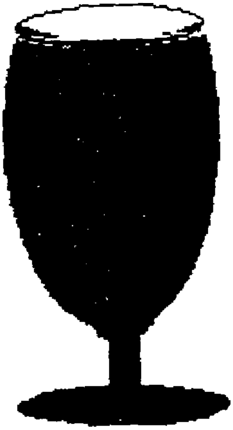
unknown = 00.
Then when doing a search, a middling person, who could be assessed also as tall or short will give a non-zero result on use of an AND mask%. In other words, there really are two questions. Could this person be thought of as tall? And, could they be thought of as short?. So you need two bits.

Database searches have lots of problems, so remember the golden rule of computing:

Would this job be faster by hand?

I'll leave you with that cheering thought. Those of you who like maths may care to think about the output produced by the following program.

```
 10 REM Binomial spread
 20 REM Run time is 104 secs
 30 DIM B%(32)
 40 FOR I% = 1 TO 1000
 50   K% = &40000000
 60   M% = RND EOR RND
 70   IF M%<0 THEN C% = 1 ELSE C% = 0
 80   FOR S% = 1 TO 31
 90     IF (M% AND K%) THEN C% = C%+1
100     K% = K% DIV 2
110     NEXT
120   B%(C%) = B%(C%)+1
130   NEXT
140
150 FOR I% = 0 TO 32
160   PRINT B%(I%);
170   NEXT
```



CLASSIFIED ADS

****

CASSETTE TAPES

C10s, boxed, leaderless - available from the UG in multiples of 20 : $26-00 plus $2-00 p&p.

MICRO COVERS

Designed to fit the BBC micro. Attractive opaque reinforced plastic, $5-50 posted. To Iain, Box 26078, Epsom, Auckland 3.

FLOPPY DISCS

3M Brand 5 1/4" DSDD discs - available from the UG ex-stock in (cardboard) boxes of 10: $55-00, which includes p&p

PROJECT: ADVENTURE INTERPRETER DEVELOPMENT SYSTEM

G.C.F.

Like every other Dungeon and Dragon fanatic, I thought the Beeb could be put to use. Level 9 games also were an inspiration, but to jump in the deep end would guarantee my death by drowning.

I plan to develop several adventure systems of progressively increasing sophistication. Each system will consist of an adventure code interpreter, which can be common to a number of different adventures, and an example adventure, to demonstrate various facilities.

Those of you who simply want to write adventures can pick on one system, and concentrate on writing your adventure without having to know much about BASIC programming. People wishing to learn new programming techniques may find the interpreter programs of considerable interest. Each interpreter will be a minimal one of its type, and will offer plenty of scope for enhancements.

The first system (VSAS) consists of 33 lines of BASIC code for the interpreter and is followed by several DATA statements specifying the actual adventure. The program plus structure diagram and other appropriate documentation will appear next month.

PROJECT OUTLINE

This is my proposed strategy, but is subject to reader interest and involvement, and my 'spare' time. Having abandoned any attempt to design an all-singing, all-dancing system straight off, I decided to approach it in six stages. This will allow experience and concepts to be progressively gained and consolidated. Each subsequent stage will be designed from first principles based on lessons learnt from previous stages, and no attempt will be made at 'forward compatibility', so that errors and awkwardnesses from previous stages will hopefully be left behind.

The six stages have been given code names for easy reference, which are partly self-descriptive of the main features. (The first 4 stages can be used either from tape or disc)

VSAS - Very Simple Adventure System:

a simple menu-driven system with no memory or conditional execution (you cannot 'get' or 'put' objects; what you can do depends only on where you are at the moment, and not on where you have been or what you have done in the past)

SAS - Simple Adventure System

This will be sufficiently advanced to provide all the facilities for a solo adventure, along the lines of many popular solo adventure books. The System can keep track of strength and skill levels as well as objects picked up. Combat routines can be specified, along with some methods of handling items with magic effects.

PAS - Parsed Adventure System

The menu system will be replaced with parsing logic to allow simple commands to be entered, such as "Put coins in box". A simple magic spell system will be implemented.

FAS - File Adventure System

Use one program to generate a file which is then read into the interpreter program. This will allow some data compaction and pre-processing to enable the interpretation to be more efficient, more sophisticated, and permit still bigger adventures.

VAS - Virtual Adventure System

This will enable a large adventure file to be set up on disc and accessed using virtual memory concepts. This will allow bigger and more sophisticated adventures to be written than by the previous methods.

ABADS - Advanced BASIC Adventure Development System

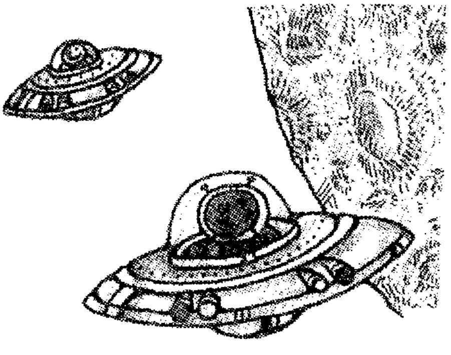
An overlayed program system which uses a fully-fledged adventure language. Players with sufficient skill will be able to modify "reality" using suitably prepared magic spells. This may be enhanced to allow a multi-player facility using several connected micros.

P.S.

I am prepared to supply tape/disc copies of programs under the following conditions:

1. You provide the tape or disc (5{" 80 track) plus return p&p
2. You are a current member of the User Group and quote your membership number
3. You also provide some constructive criticism!

So start thinking...



STUDENT DISCOUNT:

HOW IT WORKS

A school or its Computer Club may apply for membership at a reduced cost in the following way:

1. There must be 5 or more students wishing to join.
2. Each pays $20.00 instead of $30.00
3. Each student is entitled to one copy of each issue of 'Beeblet' for the year of membership.
4. All copies of 'Beeblet' will be mailed to the school where it will be the responsibility of the school or its computer club to distribute copies to the individual students.
5. The school is entitled to one vote and full rights and privileges of membership in the User Group.
6. There is no objection to one teacher being included in this membership.
7. If more pupils wish to join later in the year, they may, and back issues to April will also be supplied to them at the school.

So all you students, and teachers, you have a little time to gather more of your schoolmates to make up enough of you to qualify for this reduced rate. You can transfer your 1984/5 personal membership to the school-discount membership upon renewal time, if there are enough of you in the same school wanting to be members.

You cannot, however, renew or join as an individual then later try to transfer part way through the subscription year and get a refund.

Go to it!!! Write to us at P.O.Box 9592, Wellington, for a special Membership Application form for School groups.

BINOMIAL SPREAD ANSWER

The output from the Binomial Spread program can be predicted from the following formula:

`P(K) = 32!/(K!*(32-K)!*2^32)`

where P(K) is the probability of K bits matching. The formula assumes the two values produced by RND in line 60 are truely random. The output is a way of testing a random number generator, using the Chi-square test.

LOCAL ARRAYS

GCF

YES!! It is possible to make arrays Local to a procedure. More specifically, you can at least have parts of various arrays local. You have to declare each particular array 'cell' separately, as in the listing below.

The only problem is that there is a maximum of 255 local variables allowed for a PROC. More than this and the program hangs up. In the program 254 cells of an array plus one Integer Variable make up the maximum allowable. The example fills part of an array with consecutive integers (0,1,2,3...), and then fills the same part of the array with "9999" after making it LOCAL. Finally it demonstrates that the original values are still there.

Note that LOCAL arrays are NOT allowed on the Z80 co-processor, but are allowed on the 6502 co-processor.

```
   10 REM  To demonstrate 'LOCAL' arrays
   20 DIM AA(500)
   30 N% = 254
   40 FOR I%=0 TO N%
   50   AA(I%) = I%
   60   PRINT I%,AA(I%)
   70   NEXT I%
   80 PRINT
   90 PROCbb
  100 PRINT
  110 FOR I%=0 TO N%
  120   PRINT I%,AA(I%)
  130   NEXT I%
  140 END
  150
  160 DEF PROCbb
  170 LOCAL I%
  180 FOR I%=1 TO N%
  190   LOCAL AA(I%)
  200   AA(I%) = 9999
  210   PRINT I%;AA(I%)
  220   NEXT I%
  230 ENDPROC
```

* * *

LOCKED KEYBOARD

Type in *FX201,1, press any Key other than Break. Within a program, *FX201,0 can be used to unlock the Keyboard.

FRAK!

Reviewer Geoff P

"FRAK!" by Aardvark Software
There's no doubt about it; Scrubbly, Hooter, Poglet and Trogg are headed for immortality. They've been hailed as a milestone in graphics programming, have sat at the top of UK software charts for several months and even added a new word to the English language - FRAK!

The hero of the piece is Trogg, a bearskin-clad caveman armed with nothing more lethal than a yo-yo, and the aim of the game is to guide him around the screen collecting golden keys. Along the way he encounters diamonds and urns for valuable bonus points, must dodge vertically rising balloons and diagonally approaching daggers, yo-yo any obstructing Scrubbly, Hooter or Poglet and must simultaneously negotiate a maze of ladders and platforms. Failure to do so results in a cartoon-style voice bubble accompanied by a superbly suitable expression of frustration - FRAK!

Each section of the game is the equivalent of three screens wide and the screen scrolls effortlessly left and right as Trogg strolls about. In addition, as each section is completed, a new and completely different screen of increased complexity appears and the first problem is to find one's way around it. You can't move on until all the golden keys are collected and some of these are sited in some pretty tricky places. The third section is, understandably, the hardest with Trogg having to place his foot-steps with a precision rivalling Fred Astaire. And after that? Well, I'll leave you to find out. I'll just say that the game's programmer, the mysterious Orlando, has a deliciously wicked sense of humour!

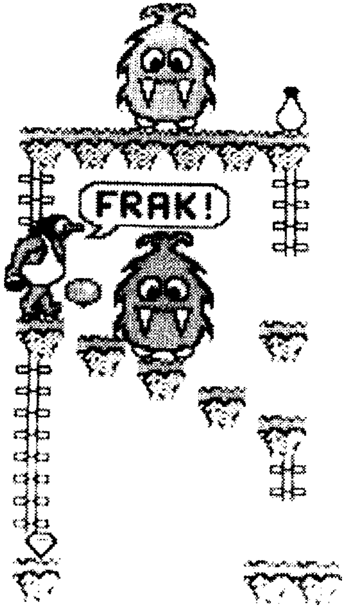
The game controls are the standard left/right, up/down format with RETURN launching the yo-yo. (One tip here; how long you press RETURN depends on how far the yo-yo travels). Jumping is achieved by the combined use of 'up' plus 'left' or 'right' and if the phone rings in the middle of a game just hit DELETE to freeze the action and COPY to restart. Each level has its own background music which, along with the, (at last!!), non-deafening sound effects can be switched on and off instantly with 'Q' and 'S'.

The one feature that stands head and shoulders above the rest in this game is the graphics. Not only does Trogg have different left and right profiles, (in keeping with a bear-skin slung across one shoulder), but you can even see his hairy chest. And he doesn't just walk, in places he positively skips along!

Another delight was the humour and originality of the game. Troggs and

Scrubblys don't die or explode when hit; the one utters his oath while the other just goes whistling off the edge of the screen. And the clock that counts down at the start of each section, and by which bonus points accrue, is merely a timer on your electricity meter. When it reaches zero there's an audible click and the lights go out! Even the first section of the program is worth a peek for the REM statements warning, tongue in cheek, of dire consequences for pirates of the game. Horsewhips, cheese-wires and Very Big Men wearing Very Big Boots are just some of them! There's even a tune for would-be pirates. The theme from 'Captain Pugwash' no less!! Type GOTO 140 for a real musical treat.

So much for the praise, now for the gripes. There's no documentation at all. You'd best grab a pencil and note down the keys to use while the game is loading because that's all the help you get. Things like the yo-yo tip mentioned above only come after a lot of frustration. Another grizzle is that uttering FRAK! sends you back to the start of whatever level you are on to face, once again, all the hazards you've just eliminated. More annoying still is that balloons and arrows passing close to Trogg will make him FRAK! without actually touching him. It's something you get used to and something I'm prepared to accept since it is apparently a side-effect of the sprite routine specially developed for the game. The other minor moan concerns the occasional disappearance of the background music. An intermittent fault this (everyone's favourite sort), that doesn't impede the progress of the game.



All in all FRAK! is a classic. It's a delight to play; it's both amusing and frustrating, is filled with subtle humour and originality and should be on every Beeb's screen!

DEALS AND DISCOUNTS

Send stamped addressed envelope for details of the deals negotiated by the Wellington Branch for UG members. Please mark your request for the attention of Jill Armstrong, C/O the UG Box 9592 Wgtn.

DISC SECTOR VERIFY

GCF

This "No Frills" program simply reads every sector on the specified drive. If an error is found it is printed out in Hex (Chapt 10 Disc U.G.), along with the track and sector numbers in decimal. (Obvious frill: insist on a valid drive number)

For more details of OSWORD with A%=&7F see end of Chapt. 8 Disc UG. Before testing (preferably on a backed up disc, or one known to be corrupt), check the way you have set up parameters etc VERY CAREFULLY. This listing is a direct printout of a working program, but no responsibility can be accepted by the author. It could form part of a disc recovery program which would try to recover as much as possible from files on a corrupt disc.

```
   10 REM  OSWRD7F    (OSWORD with A% = &7F)
   20 REM   This will check every sector on specified
drive
   30 REM  - without destroying contents.
   40 MODE6
   50 DIM instr_block% 20
   60 DIM data_area% 255
   70 INPUT "Number of drive ",D%
   80 FOR T% = 0 TO 79
   90   FOR S% = 0 TO 9
  100     E% = FN_read_sector(D%,T%,S%)
  110       IF NOT(E% = 0) THEN PRINT "ERROR &";~E%" @
Track/Sector = ";T%;"/";S%
  120     NEXT S%
  130   NEXT T%
  140 END
  150
  160 DEF FN_read_sector(drive%,track%,sector%)
  170 LOCAL A%,X%,Y%
  180 A% = &7F
  190 X% = instr_block% MOD 256
  200 Y% = instr_block% DIV 256
  210 ?instr_block% = drive%
  220 instr_block%!1 = data_area%
  230 instr_block%?5 = 3
  240 instr_block%?6 = &53
  250 instr_block%?7 = track%
  260 instr_block%?8 = sector%
  270 instr_block%?9 = &21
  280 CALL &FFF1
  290 =instr_block%?10
```

TWIN KINGDOM VALLEY By BUGBYTE SOFTWARE.

Reviewed By A. Beeston

This program is available on tape, from Software Supplies (N.Z) Ltd, P.O. Box 865, Christchurch at a cost of $34.95. Versions exist for BBC and Commodore computers.

The tape loads instructions, which provide a friendly introduction to the setting and adventure gaming on computers, then the main program is loaded.

It is an adventure game in which the adventurer aims to explore and gains points by finding treasure. This is not a true real-time adventure game. Time only passes when the player enters an instruction, and characters react only after a command is entered.

GRAPHICS

This particular program has extensive graphics of objects and locations. The graphics are very detailed and colourful. The whole screen is used for graphics therefore the program has to switch between text and graphics. The user has the choice of six options combining the graphics and text output ranging from short messages only with no graphics to long messages and graphics displayed for every location. There are locations that have no graphics as they are set in magical darkness.

FEATURES AND LIMITATIONS

You have a strength which starts at 190 and decreases (you die if it gets to zero), but can be built back up. The characters you meet in the game also are of varying strength. You meet both enemies and friends in your travels - it is up to you to identify them.

The games vocabulary is limited to movement, simple manipulation of objects and miscellaneous commands to provide information about your character and environment (score, inventory,look etc)

Once you as the player are killed the game ends, there is no provision for the player to be reincarnated. There is a facility for saving a game in progress and loading a partially completed game, so that when you die you can restart without retracing every step that you took in previous game.

My best score so far is 385 out of a total of 1024 possible points. I have a long way to go to complete this adventure.

SYSTEM INTEGER CHART

G.C.F.

This program will print out as a chart, a list of all the system integer variables within boxes on one A4 page.

WHY?

System integer variables (A% to Z%) are very precious: they are the fastest variables in Basic, they don't require additional memory, and there are only 26 of them. If your program is using a lot of them, it can be difficult to keep track of WHICH ones have been used, and also WHAT they are representing, as usually they cannot be considered as having meaningful names. So it can be vital that a record be kept of their usage, either as documentation to a very large program, or at least to prevent the same variable being used to have two different uses.

Note: the program is printer-independant except for line 60:
CHR$(&94) IS A C.Itoh printer character which is an "overline" - like an underline, but at the top. Seikoshas and Epson-alikes don't appear to have this character, but Logitec users could use the ESC Y command to define it. Otherwise, replace CHR$&94 with a hyphen - or some such.

```
 10 REM LSYSINT
 20 VDU2
 30 C% = 39
 40 D% = 78
 50 PRINT TAB(25);"System Integer Variable Usage"
 60 PRINT TAB(25);STRING$(29,CHR$(&94))
 70 PRINT
 80 PRINT TAB(28);"For Program ";STRING$(10,"_")
 90 PRINT
100 PRINT
110 FOR I% = 1 TO 13
120   PRINT STRING$(D%+1,"-")
130   PRINT " ";TAB(C%);" ";TAB(D%);" "
140   PRINT " ";SPC2;CHR$(&40+I%)+"%";TAB(C%);
150   PRINT " ";SPC2;CHR$(&40+I%+13)+"%";TAB(D%);" "
160   PRINT " ";TAB(C%);" ";TAB(D%);" "
170   NEXT I%
180 PRINT STRING$(D%+1,"-")
190 VDU12
200 VDU3
210 END
```

TRIFFID SOFTWARE RESEARCH

Reviewer Steve W

Adventures 1 (The secret river) and 2 (The wizards citadel).
Review copies supplied by TSR, Cullwood Lane, Ashley, New Milton, Hants, UK ''6.95 each plus ''1.50 airmail.

Anyone who has played "Dungeons and Dragons" will be familiar with this style of game which involves generating a "character" with particular attributes (strength, dexterity etc), and then manipulating this character through a fantasy world controlled by a Dungeon Master (in this case the computer). Unfortunately there are quite a few annoying things about the game which detract from what would be otherwise a quite playable game. Firstly, the credits refer to graphics, but this refers only to the title page (false advertising?). On the subject of the title page: this is very impressive, but how would you like to wait three minutes for 20K of title page to load from tape! The game is supplied on tape only, and loads in 5 or 6 chunks, some of which have "protected" names. This is nothing more than a nuisance as it means the novice will be unable to bypass the title screen. The credits talk about protection, but I experienced no difficulty transferring Adventure 1 to disc, whilst the second required a slight mod to run successfully from disc.

While actually playing, the response it very quick (unlike Acorn adventures!) but I personally did not like the use of colour. Another prsonal dislike was the way not all exits from a location were described. Some may prefer a bit of randomness like this as being more realistic. You do a lot of fighting in these games, which can be a drawn-out affair. Fighting in based on a system of armour-points, hit-points, and probability.

A scan through the code (all BASIC) reveals the sort of "brute force" programming that commercial programmers seem to delight in: long series of IF statements terminated by GOTOs. Very inelegant, but easy to follow (and fast!). Adventure 1 has no SAVE facility (a cardinal sin), both adventures have a number of spelling mistakes. To play Adventure 2 you MUST have Adventure 1, as the character generator program is only supplied with the first. You wouldn't get far in Adventure 2 unless you had built up your character in the first, either. The nasties are slightly harder to dispose of, but at least there is a SAVE facility.

I must admit that my personal bias for the Level 9 format for adventures may have affected my opinion of these games, but I think that there is definitely room for improvement. TSR are planning to release Adventure 3 "in the spring" (UK presumably).

FOR FAST FINGERS ONLY

Review by Geof P.

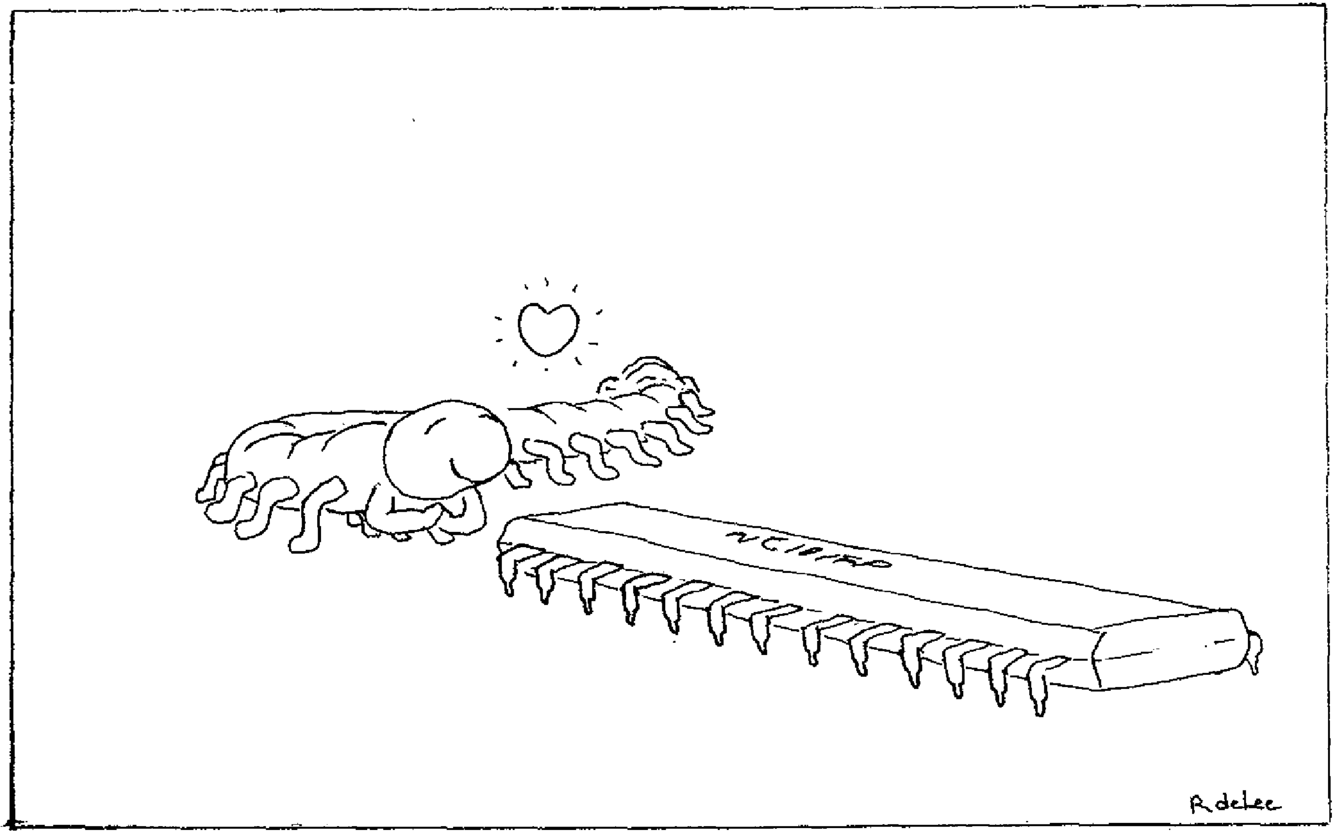
"Zalaga" Aardvark software, 100 Ardleigh Green Road, Hornchurch, Essex
RM11 2LG, England. (6.95)

Rats will apparently press a switch many hundred times a second when that switch is connected to an electrode placed in one of the pressure centres of their brains. Humans beings often exhibit similar behaviour - though for less esoteric reasons - as a quick look at ZALGA will show. Based on the arcade game of similar name its yet another of the "zap the aliens" line, and an extremely good one.

Aliens stream in from top and bottom corners, pirouette in front of your laser base then gather around the centre of the screen. Once all those that managed to get through your frantic blasting have grouped the attack proper begins with aliens descending on your ship, dropping bombs and then doing a further twirl before returning to the fleet. Occasionally one of the mother ships will whiz down and, instead of attacking, emit a long pulsed beam. Pass your ship in front of this beam and miraculously your fire power is doubled; you now have two ships side by side which means that you can do some fairly hefty damage! If you manage to hammer your way through the first two fleets there's a brief respite with a "Challeging stage" in which aliens swirl on and off the screen without dropping bombs, while you amass a bonus score. Hit all forty for 10,000 points or else its just 100 points a hit.

The graphics are extremely good with full use of sprites and very smooth movement throughout. They're also very fast! Sound too is well used without the excessive volume found in many games. You can switch sound off and on by pressing 'Q' and 'S' and there's a two player option too, again at the touch of a button. Another particular like was the use of only three keys: CAPS LOCK and CTRL for left and right and RETURN to fire - no special keys to remember at those critical moments!

A quick comparision with the arcade version shows ZALGA is even faster, which makes it quite a feat of programming. Increasing difficulty levels ensures a long interest life for the game, meaning that you are still likely to be playing it in a year's time to see if you can get a few more points or onto the next level. And I haven't even mentioned that the game has its own special typeface, or a continually scrolling background of stars. Definitely one for the collection!


ZC1070
RdeLee

HALL OF FAME

PUBLISHER:	GAME	HISCORE	SCORER
Acorn	Arcadians	21,470	Malcolm Mac.
	Aviator	5,610	Joan C's Dad
	Countdown	227	Tim R.
	Elite	Above Average	Ceri L. (age 9)
	Elite	Elite	Robert N.
	Hopper	27,196	Jenny B.
	Meteors	24,090	Duncan's flatmate
	Monsters	93,560	Linsay F.
	Planetoid	413,875	Peter J.
	Road Runner	90,900	Joanne C.
	Rocket Raid	140,777	Mark R.
	Snapper	250,000	Susan
	Starship Command	9,495	Robert N.
Alligat.	Bug Blaster	37,642	Steve W.
Aardvark	Zalaga	86,200	Malcolm Mac
	Frak	46,900	Geof P.
	Monsters	175,110	Geof P.
Bugbyte	Twin Kingdom	385	A. Beeston
Dr Soft	'747'	205	Steve W.
	Gorf	18,648	Steve W.
FG Soft.	Mutant Defender	6,900	Steve W.
Level 9	Colossal Cave	687	Steve W.
	Dungeon Adv.	330	Anne et al
	Lords of Time	925/1000	Steve W/Kerry K.
	Snowball	900/1000	Steve W./Dianne
Prog.Pow	DangerUXB(Lvl 1)	286,690	Steve W.
	Felix the Factory	7,000	Dianne S.
	Frenzy	915,468	Dianne T.
	Killer Gorilla	91,000	Susan
	Super Invaders	19,610	Brett A.
Sup.Soft	Alien Dropout	24,200	David E.
	Q*BERT	30,260	Dianne S.
	Roadruhner	70,600	Lois S.

HALL OF FAME

USER GROUP MEETINGS

AUCKLAND: -meets 2nd Wednesday of the month, 7.30pm, Conference Rooms, Auckland Secondary Teachers College,60 Epsom Ave, Epsom. ph Dave 770-630 x 510(bus) or Kerry 695-355.
CHRISTCHURCH: -fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.
DUNEDIN: -Phone Martin McDowall 42-031
HAMILTON: -last Thursday of the month 4.45pm Waikato Tech B-block staff -room. Ph. Peter, Hamilton 392-500 x897(bus) or Alison, Morrinsville 6695.
HAWKES BAY: -Alternative months Hastings & Napier. Contact Kendall Napier 435-624, Bob Taradale 446-955, Mitch Hastings 778-235.
INVERCARGILL: -Phone Evan 330-444 Invercargill.
PALMERSTON NORTH:-Phone Tom Skinner 83-009(wk)
TAUPO: -Contact Dave 84-215.
TAURANGA: -every second Wednesday. ph. Chris & Jane 65-076.
TIMARU: -Phone Lloyd van der Krogt Timaru 80-082.
TOKOROA: -meets first Friday of each month, Tokoroa High School, 8pm. Phone Brian 67-625 Tokoroa.
WELLINGTON: -meets last WEDNESDAY of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon.ph. Anton 286-289 or Warren 787-005.
OTHER CENTRES: - let us know and we will publish details here.

THE NEWSLETTER:

CONTRIBUTIONS:-most welcome. Listings should be sent in on tape ordisc.(Tape - 2 copies, one at 300 baud please). A 35c stamp gets your tape / disc returned too. Please include written explanation of listing; text only material welcome too.

DEADLINES:- For a particular issue, the last day for material is the last day of the month prior to publication.

ADVERTISING:- Rates are $20 per half page (camera ready), deadline as above.

Copyright (c) THE BBC/ACORN COMPUTER USER GROUP OF NZ INC.
Published monthly except January and mailed to financial members.

MEMBERSHIP:

Membership of the users group is on payment of an annual subscription from April to March. For the 1984/5 year it is: $30.00, plus a $5.00 joining fee for new members, plus $5.00 for Auckland & Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Renewing members get all back issues of the current year; New members also get back issues to v.1 #1 Jan '84.

THE ACORN/BBC COMPUTER USER GROUP OF NZ INC, P.O.Box 9592, Wellington.

Reproduced by ROBERTS PRINTING Co. Ltd, Wellington.